COMMUNICATIONS IN STATISTICS
Simulation and Computation®
Vol. 33, No. 2, pp. 271–292, 2004

Strategies for Efficient Computation of Multivariate Poisson Probabilities

Panagiotis Tsiamyrtzis and Dimitris Karlis*

Department of Statistics, Athens University of Economics and Business, Athens, Greece

ABSTRACT

The main reason for the limited use of multivariate discrete models is the difficulty in calculating the required probabilities. The task is usually undertaken via recursive relationships which become quite computationally demanding for high dimensions and large values. The present paper discusses efficient algorithms that make use of the recurrence relationships in a manner that reduces the computational effort and thus allow for easy and cheap calculation of the probabilities. The most common multivariate discrete distribution, the multivariate Poisson distribution is treated. Real data problems are provided to motivate the use of the proposed strategies. Extensions of our results are discussed. It is shown that probabilities, for a large family of multivariate distributions, can be computed efficiently via our algorithms.

*Correspondence: Dimitris Karlis, Department of Statistics, Athens University of Economics and Business, 76 Patission St., 10434 Athens, Greece; E-mail: karlis@aueb.gr.

DOI: 10.1081/SAC-120037235 0361-0918 (Print); 1532-4141 (Online)

Key Words: Recurrence relationships; Panjer's family of distributions; Multivariate discrete distributions.

Mathematics Subject Classifications: Primary 62E99; Secondary 62P05.

1. INTRODUCTION

The multivariate Poisson distribution, while one of the most well known and important multivariate discrete distributions, has not found a lot of practical applications apart from the special case of the bivariate Poisson distribution. The main problem that has limited the applicability of multivariate Poisson models is the awkward probability function which causes the inferential procedures to be quite complicated for practical purposes.

Despite the deep theoretical development of the multivariate Poisson distribution (see Johnson et al., 1997) there are only few applications due to the aforementioned problems. Consider the relatively simple problem of obtaining maximum likelihood (ML) estimates. In order to evaluate the likelihood, one has to calculate the probability function at all the observations. The probability function can be calculated via recurrence relationships as otherwise exhausting summations are needed. This can be quite time consuming (especially for high dimensions) and thus efficient algorithms are needed in order to facilitate the computations. Applying merely the recurrence relationships without trying to use them in an optimal way can be problematic and time demanding, too.

To further motivate our approach consider a problem with, say, 5-dimensional data. For example, the data may represent the purchase of 5 different products of a household, where each household is observed for different time periods. In fact this implies that we work with rates instead of counts, but again a multivariate Poisson model seems plausible. If the number of households is not very large, this implies that we will have a large number of cells with zero frequency, so the calculation of the entire 5-dimensional space of all combinations for the number of purchases of the five products is a very bad strategy. For instance, if the maximum number of purchases for each product is denoted as z_i, then according to the usual strategy of creating the entire probability table, one has to calculate $\prod_{i=1}^{5}(z_i+1)$ different probabilities. Moreover, if one works with rates rather than counts, this implies that the parameters for each individual are different and such a table must be created for each observation. Clearly, calculating all these probabilities is awkward and time consuming. We would be

pleased if we had to calculate fewer probabilities and especially only those for non-zero frequency cells which contribute to the likelihood. This simplified example shows that efficient algorithms for the computation of probabilities can be quite helpful for improving the applicability of the model.

The purpose of the present paper is the proposal of efficient strategies for calculating the probabilities based on the existing recurrence relationships. The proposed approach can reduce considerably the computing time especially for large dimensions and/or high values of the underlying variables.

The remaining of the paper proceeds as follows. In Sec. 2 we introduce briefly the multivariate Poisson distribution and the recurrence relationships available, while at Sec. 3 we discuss the bivariate case. The bivariate case, offers interesting geometric interpretation of the proposed algorithms, adding insight to our approach. The algorithms are generalized in Sec. 4 for higher dimensions. In Sec. 5 we discuss the practical usefulness of the proposed strategy exploiting the gain in computing effort by using our approach. Section 6 discusses the potential generalization of our approach to a variety of other bivariate distributions and their multivariate extensions. Finally, concluding remarks can be found in Sec. 7.

2. THE MULTIVARIATE POISSON DISTRIBUTION

The multivariate reduction technique has been used to create the multivariate Poisson distribution. This technique has been used extensively for the construction of multivariate models (see e.g., Mardia, 1971). The idea is to start with some independent random variables (which are generally elementary) and to create new ones by considering some functions of the original variables. Then, since the new variables contain jointly some of the original ones, a kind of structure is imposed creating multivariate models.

Suppose that Y_i are independent Poisson random variables with mean θ_i, for $i = 0, \ldots, n$ and let $X_i = Y_0 + Y_i$, $i = 1, \ldots, n$. Then the random vector $\mathbf{X} = (X_1, X_2, \ldots, X_n)$ follows a n-variate Poisson distribution, where n denotes the dimension of the distribution.

The joint probability function is given by

$$P(\mathbf{X} = \mathbf{x}) = P(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n)$$
$$= \exp\left(-\sum_{i=0}^{n} \theta_i\right) \prod_{i=1}^{n} \frac{\theta_i^{x_i}}{x_i!} \sum_{i=0}^{s} \left[\prod_{j=1}^{n} \binom{x_j}{i} \right] i! \left(\frac{\theta_0}{\prod_{k=1}^{n} \theta_k} \right)^i \qquad (2.1)$$

where $s = \min\{x_1, x_2, \ldots, x_n\}$. This distribution is denoted as $n - P(\boldsymbol{\theta})$, where $\boldsymbol{\theta} = (\theta_0, \ldots, \theta_n)$. Marginally each X_i follows a Poisson distribution with parameter $\theta_0 + \theta_i$. Parameter θ_0 is the covariance between all the pairs of random variables. If $\theta_0 = 0$ then the variables are independent and the multivariate Poisson distribution reduces to the product of independent Poisson distributions.

The above defined version of multivariate Poisson distribution is a reduced version of a more general model used by many authors (Johnson et al., 1997; Loukas and Kemp, 1983; Mahamunulu, 1967) which assumes another more complicated structure. However, this reduced version is the only one used for real applications. The literature for the multivariate Poisson distribution is large and many references, as well as historical remarks, can be found in Johnson et al. (1997) and Krummenauer (1998). The bivariate Poisson distribution has been studied in more depth. The reader can refer to the book of Kocherlakota and Kocherlakota (1992).

As mentioned, the main obstacle which limits the usage of discrete multivariate distributions in practice, is the complexity of calculating the probability function. The summations needed might be exhausting in some cases especially when the dimension is large. Computation of the probabilities can be accomplished via recursive schemes. Kano and Kawamura (1991) provided a general scheme for constructing recurrence relations for multivariate Poisson distributions.

Let us introduce some notation to facilitate the exposition. Let $\mathbf{0}$, $\mathbf{1}$ denote the vector with all elements equal 0 and 1 respectively and $\mathbf{e_i}$ the vector with all elements 0 except from the ith element which is equal to 1. Using this notation, we may define a recursive scheme for the multivariate Poisson distribution defined in (2.1) using the relationships:

$$x_i P(\mathbf{X}) = \theta_i P(\mathbf{X} - \mathbf{e_i}) + \theta_0 P(\mathbf{X} - \mathbf{1}), \quad i = 1, \ldots, n \tag{2.2}$$

$$P(X_1 = x_1, \ldots, X_k = x_k, 0, 0, \ldots, 0) = P\left(\mathbf{X} - \sum_{i=1}^{k} \mathbf{e_i}\right) \prod_{i=1}^{k} \frac{\theta_i}{x_i} \tag{2.3}$$

for $k = 1, \ldots, n-1$, where the order of X_i's and 0's can be interchanged to cover all possible cases, while $P(\mathbf{0}) = \exp(-\sum_{i=0}^{n} \theta_i)$.

It can be seen that, since at every case at least one of the x_i's equals 0, i.e., $s = 0$, the sum appearing in the joint probability function has just one term and hence the joint probability function takes the useful form $P(\mathbf{X}) = \exp(-\theta_0) \prod_{i=1}^{n} Po(x_i \mid \theta_i)$, where $Po(x \mid \theta) = \exp(-\theta)\theta^x / x!$ denotes the probability function of the simple Poisson distribution with parameter θ. Then (2.3) arises by using the recurrence relation for the univariate Poisson distribution.

It is clear that as n increases the computational effort increases, too. In fact for a n-variate Poisson model one has to initialize in this way $\sum_{i=1}^{n-1} \binom{n}{i}$ probabilities of the form $P(X_1 = x_1, \ldots, X_k = x_k, 0, 0, \ldots, 0)$, for $k = 1, \ldots, n-1$ interchanging the order of zeroes and x_i's. For example, if $n = 4$, 14 different recursive schemes must be run. The computational burden is large. However, the use of the above defined recursions seems to be the only way to calculate the probability function.

In the next section we will treat the bivariate model in order to give insight into the proposed algorithms and improve the understanding of the problem treated in this paper.

3. THE BIVARIATE CASE

Consider the recurrence relationships defined in (2.2). The general idea for calculating the probabilities is to start from $P(\mathbf{0})$ and then obtain all the other probabilities via the recursive schemes defined above. The question that naturally arises is which one of the relationships one has to use and/or can we improve the computing time needed by combining the relationships? Moreover, is it always true that combining, somehow, the relationships would lead to a more efficient algorithm? These are the questions that we will try to answer in the sequel. At this section we will discuss the simplest but intuitive case, that of a bivariate model.

3.1. Graphical Presentation

The bivariate Poisson distribution has joint probability function given by:

$$P(x, y) = P(X = x, Y = y)$$
$$= e^{-(\theta_0+\theta_1+\theta_2)} \frac{\theta_1^x}{x!} \frac{\theta_2^y}{y!} \sum_{i=0}^{s} \binom{x}{i} \binom{y}{i} i! \left(\frac{\theta_0}{\theta_1\theta_2}\right)^i,$$

where $s = \min\{x, y\}$. According to the general recurrence in (2.2) we have that:

$$\begin{aligned} xP(x, y) &= \theta_1 P(x-1, y) + \theta_0 P(x-1, y-1) \\ yP(x, y) &= \theta_2 P(x, y-1) + \theta_0 P(x-1, y-1). \end{aligned} \tag{3.1}$$

with the convention that $P(x, y) = 0$, if $s < 0$.

The $P(0, 0)$ probability can be obtained directly without special effort and the other probabilities will be derived from it. The key idea is

MARCEL DEKKER, INC.
270 Madison Avenue, New York, New York 10016

that if one of the coordinate values is 0, then the recurrence relationship is simpler and thus it seems reasonable to try to move towards to one of the axes. Suppose that we are interested in calculating the probability at (4, 5). One may use only one recurrence relationship to calculate the probability $P(X = 4, Y = 5)$. Figures 1a and 1b depict the entire paths and the probabilities that one must compute in order to obtain the requisite probability by using only one of the relationships. The arrows indicate the probabilities needed in order to calculate each point. It is

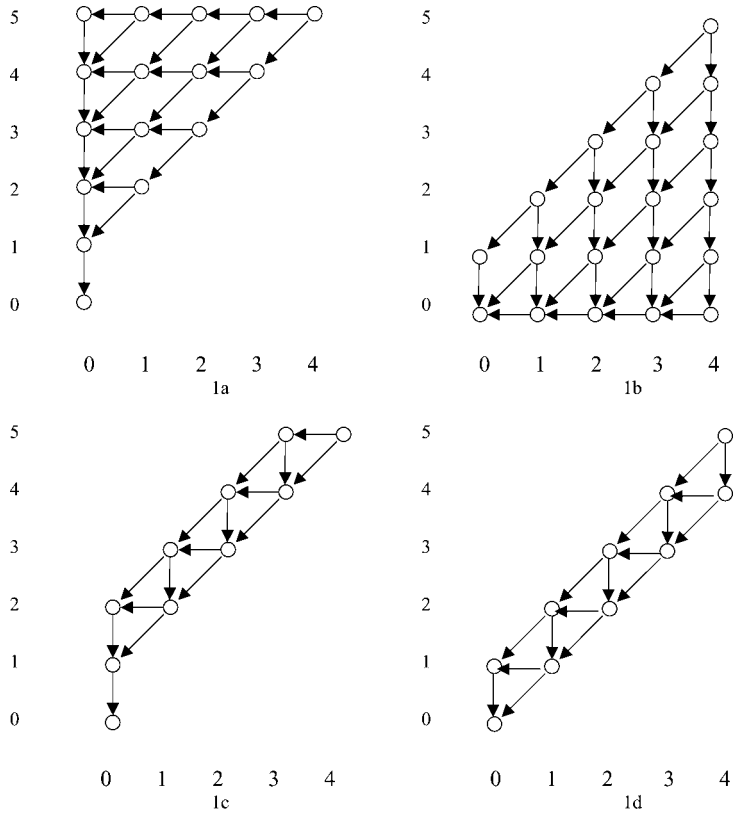

Figure 1. Four different approaches for calculating the probability at the point (4, 5). The first two (a and b) make use of only one of the recurrence relationships, while the other figures (c and d) make use of both recurrence relationships. The arrows indicate the points needed in order to obtain a specific point. It is clear that if we use a combination of the two relationships we need less points.

Marcel Dekker, Inc.
270 Madison Avenue, New York, New York 10016

clear that by using only one recurrence relationship we calculate much more probabilities than what we really need.

More formally, assume that we are interested in calculating $P(x, y)$ where without loss of generality $x \leq y$. Geometrically, using only the first of the recurrences in (3.1) we obtain the entire triangle defined by the points (x, y), $(0, y)$ and $(0, y - x)$ until to get to the y axis (where we can use the simplified recurrences in (2.3) to move to the (0, 0) point). On the other hand if we had used only the second of the recurrences in (3.1) then we would obtain the trapezoid defined by (x, y), $(x, 0)$, $(0, 0)$ and $(0, y - x)$.

Consider now the case where we may combine the two recurrence relationships. Figures 1c and 1d describe an algorithm where the two relationships are used in a successive manner (in 1c we used firstly the relationship which "moves" towards to the x axis and then the other one, while in 1d we did the opposite). Despite the fact that both of these paths gave the same number of points needed to estimate $P(4, 5)$ we will prefer the first one since it contains 3 points where at least one of the coordinates is 0 in contrast to 1d where there are 2 such points. Therefore it is clear from these figures that combining the relationships in the right order reduces the number of points needed. This reduction can be dramatic for higher dimensions and for large x and/or y. However, it must be pointed out that the situation is much more complicated for higher dimensions as we will see in Sec. 4.

Concluding, we have seen graphically that combining the two relationships one can improve the speed for calculating probabilities.

3.2. The Proposed Algorithm

We are interested in calculating $P(x, y)$ and assume that $x \leq y$. We will use the recurrence relationships (3.1) to move towards (0, 0). Given that both the recurrence relationships contain the point $(x - 1, y - 1)$ after x uses we will hit the y axis at $(0, y - x)$. Thus it is wise to use the recurrence relationships in such a way that we will stay as close to the diagonal connecting the points (x, y), $(0, y - x)$ as possible. This can be done by using both the recurrence relationships in an alternating order, since in this way after the first use every time we will move to a new point and an existing one. More specifically: from (x, y) we move to $(x - 1, y)$ and $(x - 1, y - 1)$, then applying the other recurrence relationship on the point $(x - 1, y)$ we get again the $(x - 1, y - 1)$ and the point $(x - 2, y - 1)$. Thus from (x, y) after a full use of the recurrence relationships we get to to the $(x - 1, y - 1)$ and $(x - 2, y - 1)$ points, having

used one intermediate point, the point $(x-1, y)$. So if we call a *step* the use of both the recurrence relationships in the order: first the one which moves to the closest axis and second the other one, then at the end of every but the first step we have introduced two new points (in the first step we have introduced an extra intermediate point not being used later).

Thus, in order to stay as close as possible to the diagonal parallel to the first diagonal of the axes we need to use both the recurrence relationships. Given that we are interested in intersecting an axis as soon as possible (because afterwards we can move faster to the (0, 0) point) we will do the first move parallel to the axis with the minimum coordinate. Thus starting from (x, y) after $x-1$ steps we will have the points $(0, y-x+1)$ and $(1, y-x+1)$. Then we will use only the first recurrence relationship of a full step to move from $(1, y-x+1)$ to $(0, y-x+1)$ and $(0, y-x)$. So, after $2(x-1)+1$ uses of the recurrence relationships we passed through $2(x-1)+2=2x$ points where the last 2 points are on the maximum coordinate axis: $(0, y-x+1)$ and $(0, y-x)$. Then we can move from the $(0, y-x+1)$ to the (0, 0) point using $y-x+1$ times the recurrence relationship (2.3). So the proposed algorithm to calculate the $P(x, y)$ is the following:

3.3. Algorithm

Find the minimum coordinate (assume $x \le y$).
Calculate $P(0, 0)$
FOR $k = 1$ TO $y-x+1$

$$P(0,k) = \frac{\theta_2}{k} P(0, k-1)$$

END
FOR $k = 1$ TO $x-1$

$$P(k, y-x+k) = \frac{\theta_1}{k} P(k-1, y-x+k) + \frac{\theta_0}{k} P(k-1, y-x+k-1)$$

$$P(k, y-x+k+1) = \frac{\theta_2}{y-x+k+1} P(k, y-x+k) + \frac{\theta_0}{y-x+k+1} P(k-1, y-x+k)$$

END

$$P(x,y) = \frac{\theta_1}{x} P(x-1, y) + \frac{\theta_0}{x} P(x-1, y-1)$$

Marcel Dekker, Inc.
270 Madison Avenue, New York, New York 10016

Remark. If $s = 1$ there is no need to combine the relationships, as the minimum coordinate will be zeroed after the first step.

4. THE ALGORITHMS FOR THE *n*-VARIATE POISSON DISTRIBUTION

While the bivariate case gave interesting graphical representation, in more dimensions the problem is much different. In the sequel we will provide two different approaches. The former one uses all n recurrence relationships while the latter uses only one of them. We will start with what we will call the full algorithm to emphasize that it uses all n recurrence relationships.

4.1. The Full Algorithm

We are interested in calculating the probability $P(x_1, \ldots, x_n)$ from a n-variate Poisson. Without loss of generality, assume that $x_1 \leq x_2 \leq \cdots \leq x_n$. The algorithm that we propose consists of two stages. In the first stage using the recurrence relationships (2.2) we will try to move from the $\mathbf{X}$ point to the closest hyperplane where at least one of the coordinates becomes 0 and then at the second stage use of the simplified recurrence relationships (2.3) will get us to the $\mathbf{0}$ point.

Stage 1. Since all the recurrence relationships (2.2) contain the point $\mathbf{X} - \mathbf{1}$, no matter which ones and in what order they will be used we will move along the "diagonal" defined by the points $\mathbf{X}$ and $\mathbf{X} - \mathbf{1}$ towards the hyperplane where the x_1 coordinate becomes 0. Thus we will try to make use of (2.2) in such a fashion so as to stay as close as we can to this "diagonal". This can be done adopting the scheme presented in Fig. 2.

Starting from the point $\mathbf{X}$ we use firstly the recurrence relationship which moves parallel to the axis of x_1 (the minimum coordinate). This will give two new points, one on the diagonal $\mathbf{X} - \mathbf{1}$ and a second one. We leave the diagonal point as it is and we break the other one using the recurrence relationship which moves parallel to the x_2 axis. We repeat this break up of the non-diagonal elements using every time the recurrence relationship of the next ordered coordinate. At the nth break up we get a new diagonal point and the $\mathbf{X} - \mathbf{1}$ as the non-diagonal. If we call a *step* the use of all recurrence relationships in the above described order, then once a *step* is completed we have moved to n new points (in the first step we have passed also through $n - 1$ extra intermediate

Marcel Dekker, Inc.
270 Madison Avenue, New York, New York 10016

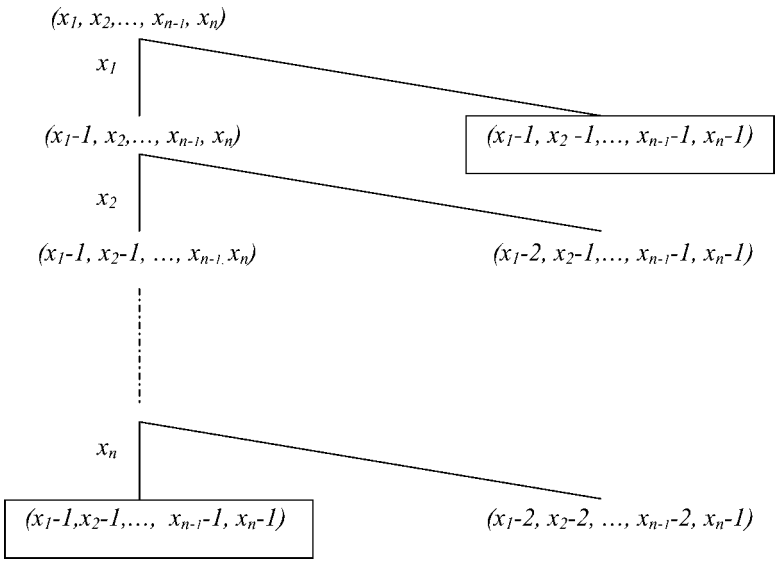

Figure 2. The path of a complete *step* in the general n-variate Poisson distribution when all the recurrence relationships are used (full algorithm). It is evident that after completing one step we end up with exactly n points since there is always one point that it is duplicated.

points; the non-diagonal elements). The new n points that we have been moved to, once the first step is completed, are the following: $\mathbf{X}-\mathbf{1}$, $\mathbf{X}-\mathbf{1}-\mathbf{e_1},\ldots,\mathbf{X}-\mathbf{1}-\sum_{k=1}^{n-1}\mathbf{e_k}$.

Starting again the break up from the $\mathbf{X}-\mathbf{1}$ point and following the same strategy as the one described above, upon completion of the second step, we will have defined n new points (here there are no intermediate points since for every break up we get a new diagonal element and a non diagonal which is one of the diagonal of the previous step). The n new points at the end of the second step are: $\mathbf{X}-2\times\mathbf{1},\mathbf{X}-2\times\mathbf{1}-\mathbf{e_1},\ldots,\mathbf{X}-2\times\mathbf{1}-\sum_{k=1}^{n-1}\mathbf{e_k}$.

Therefore after x_1-1 steps we will have the following n points, where all but the first one belong to the hyperplane defined by the coordinate of x_1 being 0: $\mathbf{X}-(x_1-1)\mathbf{1},\mathbf{X}-(x_1-1)\mathbf{1}-\mathbf{e_1},\ldots,$ $\mathbf{X}-(x_1-1)\mathbf{1}-\sum_{k=1}^{n-1}\mathbf{e_k}$.

In order to complete the Stage 1 of our strategy we will break the first point of the last step using the recurrence relationship involving the x_1 coordinate which will give the $\mathbf{X}-(x_1-1)\mathbf{1}-\mathbf{e_1}$ (already existing) and the $\mathbf{X}-x_1\mathbf{1}$ point. Thus after x_1-1 complete steps we need to calculate $1+(n-1)+n(x_1-1)+1=nx_1+1$ points in order to move

Marcel Dekker, Inc.
270 Madison Avenue, New York, New York 10016

from the $\mathbf{X}$ point to the n points over the closest hyperplane of dimension $n-1$. Note that the above formula holds only for $x_1 > 1$, since if $x_1 \leq 1$ we can move to the axis by just one move.

Stage 2. At the end of the first stage of our strategy we have moved from $\mathbf{X}$ to n points over the hyperplane of dimension $n-1$ where all have 0 in the spot of the x_1 coordinate, namely the points $\mathbf{X} - x_1\mathbf{1}$, $\mathbf{X} - (x_1 - 1)\mathbf{1} - \mathbf{e_1}, \ldots, \mathbf{X} - (x_1 - 1)\mathbf{1} - \sum_{k=1}^{n-1} \mathbf{e_k}$. From all these n points the one closest (with respect to the Euclidean norm) to $\mathbf{0}$ is the $\mathbf{X} - x_1\mathbf{1}$ point. Using (2.3) we will map the rest $n-1$ points to this point. Then from this point we will move "diagonally" down to $\mathbf{0}$. More specifically using (2.3), $x_2 - x_1$ times we will move from $(0, x_2 - x_1, \ldots, x_n - x_1)$ to $(0, 0, x_3 - x_2, \ldots, x_n - x_2)$, i.e., we will move to the hyperplane of dimension $n-2$ where in the spot of both x_1 and x_2 coordinate we have 0's. Working similarly we will get after $(x_n - x_{n-1}) + (x_{n-1} - x_{n-2}) + \cdots + (x_2 - x_1) = x_n - x_1$ steps to the $\mathbf{0}$ point.

Summarizing we observe that following this strategy in order to move from the point $\mathbf{X}$ to $\mathbf{0}$ we need to go through $nx_1 + 1$ points in stage 1 and $x_n - x_1$ points in stage 2. Thus we need:$(n-1)x_1 + x_n + 1$ points to calculate. A formal description of the full algorithm can be found in the Appendix 1.

Next we will turn to the algorithm which makes use of only one recurrence relationship that we will call the flat algorithm.

4.2. The Flat Algorithm

Similarly as before we will calculate $P(\mathbf{X})$ in two stages. In the first one we will move from $\mathbf{X}$ to the closest hyperplane using only one of the recurrence relationships (2.2) and in the second stage we will move down to the $\mathbf{0}$ point by the simplified recurrence relationships (2.3).

Stage 1. Since we need to move to the closest hyperplane using only one recurrence relationship we will use the one which moves in direction "parallel" to the minimum coordinate axis, i.e.,

$$x_1 P(\mathbf{X}) = \theta_1 P(\mathbf{X} - \mathbf{e_1}) + \theta_0 P(\mathbf{X} - \mathbf{1})$$

Thus starting from $\mathbf{X}$ and applying the recurrence relationship we get the two new points $\mathbf{X} - \mathbf{e_1}$ and $\mathbf{X} - \mathbf{1}$. Then applying the same recurrence relationship to these 2 points we are getting 3 new points: $\mathbf{X} - 2\mathbf{e_1}$, $\mathbf{X} - \mathbf{e_1} - \mathbf{1}$ and $\mathbf{X} - 2 \times \mathbf{1}$. Figure 3 shows how this break up continues until we reach the closest hyperplane defined by $x_1 = 0$.

Marcel Dekker, Inc.
270 Madison Avenue, New York, New York 10016

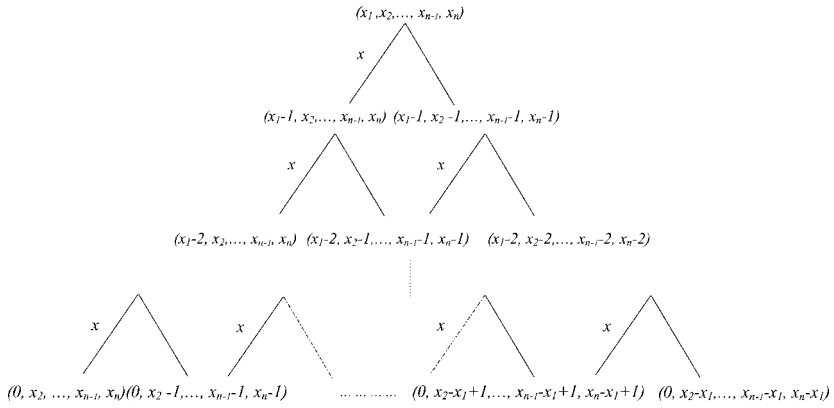

Figure 3. The path till the minimum coordinate is zeroed for the case of the flat algorithm. In every level the number of points is one more than those of the previous level.

If we call a *step* the use of the recurrence relationship to k points then upon completion of this step we will get $k+1$ new points. Thus we will reach the closest hyperplane after x_1 steps having gone through $1+2+\cdots+(x_1+1)=(x_1+1)(x_1+2)/2$ points. The x_1+1 points lying over the closest hyperplane are: $\mathbf{X}-x_1\mathbf{e_1}, \mathbf{X}-x_1\mathbf{e_1}-\sum_{i=2}^{n}\mathbf{e_i}, \mathbf{X}-x_1\mathbf{e_1}-2\sum_{i=2}^{n}\mathbf{e_i}, \ldots, \mathbf{X}-x_1\mathbf{e_1}-x_1\sum_{i=2}^{n}\mathbf{e_i}=\mathbf{X}-x_1\mathbf{1}$. Note that all these points lying over the line connecting $\mathbf{X}-x_1\mathbf{e_1}$ and $\mathbf{X}-x_1\mathbf{1}$.

Stage 2. At the second stage where we have been moved to the closest hyperplane we get x_1+1 points with the point $\mathbf{X}-x_1\mathbf{1}$ being the one closest to $\mathbf{0}$. We will move from this point all the way down to the beginning of the axes in the exact same way as we did for the full algorithm. Therefore we need x_n-x_1 extra points.

Summarizing with this algorithm we need $(x_1+1)(x_1+2)/2$ points to reach the closest hyperplane and x_n-x_1 to get to $\mathbf{0}$. Therefore the flat algorithm requires $(x_1+1)(x_1+2)/2+(x_n-x_1)$ points. A formal description of the algorithm can be found in Appendix 2.

4.3. Comparing the Flat and the Full Algorithms

Since both the algorithms act the same way at stage 2 where the closest hyperplane has been reached we will compare only the first stage of these two algorithms. What is the philosophy hidden behind these

two algorithms? In the flat algorithm we move from the point $\mathbf{X}$ to the closest hyperplane by moving along a subspace of dimension 2 (plane), independently of the dimension of $\mathbf{X}$. On the other hand the full algorithm tries to stay as close as possible to the main diagonal (defined by the points $\mathbf{X}$ and $\mathbf{X} - x_1\mathbf{1}$) by "travelling" through all the dimensions and moving a n-polytope along the main diagonal until it reaches the closest hyperplane. In the following Lemma we will see under what conditions we should prefer the one over the other algorithm.

Lemma. *When we have to calculate the n-variate Poisson probability* $P(\mathbf{X}) = P(x_1, \ldots, x_n)$ *then*:

(i) *If* $\min\{x_i\} < 2n - 3$, *use the flat algorithm.*
(ii) *If* $\min\{x_i\} > 2n - 3$, *use the full algorithm.*
(iii) *If* $\min\{x_i\} = 2n - 3$, *it is indifferent which algorithm you will use.*

Proof. If $\min\{x_i\} \leq 1$ then clearly the two algorithms are equivalent. Assume that $\min\{x_i\} > 1$. Since both the algorithms behave the same way once they get to the closest hyperplane (i.e., both require $\max\{x_i\} - \min\{x_i\}$ points to get to $\mathbf{0}$) we will compare the number of points needed from each algorithm to reach from $\mathbf{X}$ to the closest hyperplane. If we will call p and p' the number of points required by the flat and the full algorithms respectively and $s = \min\{x_i\}$ we have $p = (s+1) \times (s+2)/2$ and $p' = ns + 1$. Thus we have

$$p - p' = \frac{1}{2}s^2 + \frac{3}{2}s + 1 - ns - 1 = \frac{1}{2}s(s + 3 - 2n)$$

Therefore in the case where $s = \min\{x_i\} = 2n - 3$ we get $p = p'$ and both algorithms will require the exact same amount of points to reach the closest hyperplane. If $\min\{x_i\} < 2n - 3 (\min\{x_i\} > 2n - 3)$ the flat (full) algorithm should be used since $p < p' (p > p')$.

Table 1 presents the value of the minimum coordinate for a range of different dimensions n, in order the flat algorithm to be more efficient. Clearly for $n = 2$ the flat algorithm is always inferior, while for $n = 3$ only if $\min\{x_i\} = 2$ the flat algorithm is preferable. For higher dimensions as one can see from Table 1 the flat algorithm is preferable

Table 1. The trade-off between dimension and minimum coordinate for superiority of the flat algorithm.

n	2	3	4	5	8	10	12	15	20
$\min\{x_i\}$	0	2	4	6	12	16	20	26	36

MARCEL DEKKER, INC.
270 Madison Avenue, New York, New York 10016

(for instance if $n = 10$ then $\min\{x_i\}$ needs to be higher than 16 in order to prefer the full algorithm). Note that one may combine the two algorithms for creating more efficient algorithms but this is beyond the scope of the present paper.

5. EXAMPLES AND ILLUSTRATIONS

5.1. Application: ML Estimation

In a recent paper, Karlis (2003) proposed ML estimation for the n-variate Poisson distribution via an EM scheme. Consider the more general case, that of rates rather than simple counts. Now the vector of observations $\mathbf{X}_i = (X_{1i}, X_{2i}, \ldots, X_{ni})$, $i = 1, 2, \ldots, N$ follows a n-variate Poisson distribution with parameter vector $\boldsymbol{\theta} t_i$, denoted as $P(\mathbf{X}_i; \boldsymbol{\theta} t_i)$, where $\boldsymbol{\theta} = (\theta_0, \theta_1, \ldots, \theta_n)$ and t_i can represent time, area, etc. This seems a plausible model for certain practical situations with count data. For example if the data are the occurrences of different type of diseases in different areas, then t_i's could be the populations or the area sizes. In marketing research if the data consists of the purchases of different products the t_i's could be different observational periods etc.

The loglikelihood takes the complicated form

$$L = \prod_{i=1}^{N} P(\mathbf{X}_i; \boldsymbol{\theta} t_i)$$

and numerical methods are needed. No matter what numerical method will be chosen, we need to calculate the probabilities several times, with different parameters each time. Note also that, since every observation has a different t_i, the parameters of the n-variate distribution are different for each observation. Thus, even for the calculation of the loglikelihood, we have to compute all these probabilities and thus we need efficient ways to do so.

Let us go back to the EM approach proposed. The description of the algorithm is the following:

E-step: Using the data and the current estimates after the kth iteration $\boldsymbol{\theta}^{(k)}$, calculate the pseudo-values

$$s_i = E(Y_{0i} \mid \mathbf{X}_i, t_i, \boldsymbol{\theta}^{(k)}) = \theta_0 t_i \frac{P(\mathbf{X}_i - \mathbf{1})}{P(\mathbf{X}_i)}, \quad i = 1, \ldots, N$$

M-step: Update the estimates by ${\theta_0}^{(k+1)} = \sum_{i=1}^{N} s_i / \sum_{i=1}^{N} t_i$ and ${\theta_j}^{(k+1)} = \bar{x}_j / \bar{t} - {\theta_0}^{(k+1)}$, $j = 1, \ldots, n$.

MARCEL DEKKER, INC.
270 Madison Avenue, New York, New York 10016

If some convergence criterion is satisfied stop iterating otherwise go back to the *E-step* for one more iteration.

Looking at the *E-step* it is obvious that a large number of probabilities is required in every iteration. We are not interested in calculating the entire probability table but merely two probabilities for each observation. Taking into account the fact that we need a lot of iterations, i.e., we have to calculate a lot of probabilities in a repetitive manner, it is clear that efficient and quick algorithms for obtaining the probabilities are of special importance.

5.2. Numerical Comparison

The purpose of this section is to present the number of probabilities needed for several different cases of varying dimensions. It must be noted that the recurrence relationships defined in (2.2) are based on two multiplications and one addition while the simplest recurrence (2.3) (i.e., when at least one of the coordinates is 0) is based on one multiplication only. We will compare the number of different probabilities that we need for the calculation of the target probability. Our aim is two fold: the first one has to do with the feasibility of the calculations, we believe that using efficiently the recurrence relationships the computational burden is reduced and thus the model is applicable in many situations without special effort, while the second point has to do with the comparison of the two algorithms and a general guidance for their use.

In Table 2 one can see the number of probabilities need to be calculated for various dimensions, using the two algorithms. In fact, since the number of points that we need to calculate is a function of the minimum and the maximum coordinates, we present the fewest number of points needed (attainable by any of the two algorithms). The purpose of this table is to show that even for problems that seem quite complicated, as for example when dimension $n = 10$, the number of computations needed is not prohibitively large and thus even such complicated models can be used in practice without considerable effort.

6. EXTENSIONS

6.1. General Multivariate Poisson Models

The proposed algorithms can be extended to two different avenues. First, one may consider a more general multivariate Poisson that allows

Table 2. The total number of points need to be calculated for various configurations (a or/and b indicate whether the Flat or/and Full Algorithms respectively achieve the minimum number of points). As one can see even for complicated cases with large dimension the number of calculation is relatively small.

Dimension	Min	Max	Number of points	Dimension	Min	Max	Number of points
2	0	5	6 ab	5	0	5	6 ab
2	0	10	11 ab	5	0	10	11 ab
2	5	10	16 b	5	5	10	26 a
2	5	15	21 b	5	5	15	31 a
2	10	15	26 b	5	10	15	56 b
2	10	20	31 b	5	10	20	61 b
3	0	5	6 ab	10	0	5	6 ab
3	0	10	11 ab	10	0	10	11 ab
3	5	10	21 b	10	5	10	26 a
3	5	15	26 b	10	5	15	31 a
3	10	15	36 b	10	10	15	71 a
3	10	20	41 b	10	10	20	76 a

full structure (see, e.g., Mahamunulu, 1967). Its derivation is based on a general multivariate reduction scheme where there are terms for all the pairwise covariances, covariances among 3 variables and so on.

Assume, that $Y_i, i = 1, \ldots, k$ are independent Poisson random variables and $\mathbf{A}$ is a $n \times k$ matrix with zeros and ones. Then the vector $X = (X_1, X_2, \ldots, X_n)$ defined as $X = \mathbf{A}Y$ follows a n-variate Poisson distribution.

The most general form assumes that $\mathbf{A}$ is a matrix of size $n \times (2^n - 1)$ of the form

$$\mathbf{A} = [A_1, A_2, A_3, \ldots, A_n]$$

where A_i, $i = 1, \ldots, n$ are matrices with n rows and $\binom{n}{i}$ columns. The matrix A_i contains columns with exactly i ones and $n - i$ zeros, with no duplicate columns, for $i = 1, 2, \ldots, n$. Thus A_n is the column vector of 1's while A_1 becomes the identity matrix of size $n \times n$.

For example, the n-variate Poisson distribution defined in Sec. 2 uses $\mathbf{A} = [A_1, A_n]$.

The fully structured multivariate Poisson model has not found any real data applications for two reasons. The first has to do with the complicated form of the probability function that seems to be an unsurmountable problem. The second reason is that it imposes too much structure.

Marcel Dekker, Inc.
270 Madison Avenue, New York, New York 10016

Kawamura (1985) and Kano and Kawamura (1991) described recurrence relations for this general form. Clever combination of the recurrence relations can help in obtaining the probabilities without spending much of computing time, especially for large dimensions. This would lead to the easier applicability of such complicated models.

6.2. Other Multivariate Models Defined via Recurrence Relationships

The second way to generalize the findings has to do with other families of multivariate discrete distributions that possess similar recurrence relationships. It is important to note that there are several families of bivariate distributions defined via their recurrence relationships, generalizing existing results for the univariate case.

For example, Panjer (1981) examined the family for distributions defined via recurrence relations of the form

$$P(K=k)=\left(\alpha+\frac{\beta}{k}\right)P(K=k-1),\quad k\geq 1,\ \ \alpha,\beta\geq 0$$

denoted as $K\sim\mathscr{R}(\alpha,\beta)$. It is important to note that this family contains the well known Poisson, geometric, binomial and negative binomial distributions. Hesselager (1996) defined a bivariate distribution using trivariate reduction of variates from this family of distributions. Namely he gave recurrence relationships for bivariate distributions defined as $X=R_0+R_1$ and $Y=R_0+R_2$ where $R_j\sim\mathscr{R}(\alpha_j,\beta_j)$, $j=0,1,2$ and they are mutually independent.

Then the resulting bivariate distribution has recurrence relationships of the form

$$\begin{aligned}P(x,y)&=\left(\alpha_0+\frac{\beta_0}{x}\right)P(x-1,y-1)+\left(\alpha_1+\frac{\beta_1}{x}\right)P(x-1,y)\\&\quad+\left(\alpha_0\alpha_1+\frac{\alpha_1\beta_0+\alpha_0\beta_1}{x}\right)P(x-2,y-1),\quad x\geq 1,\\P(x,y)&=\left(\alpha_0+\frac{\beta_0}{y}\right)P(x-1,y-1)+\left(\alpha_2+\frac{\beta_2}{y}\right)P(x,y-1)\\&\quad+\left(\alpha_0\alpha_2+\frac{\alpha_2\beta_0+\alpha_0\beta_2}{y}\right)P(x-1,y-2),\quad y\geq 1.\end{aligned}$$

respectively, assuming that $P(x,y)=0$ if $\min\{x,y\}<0$.

One can easily verify that using only a single relationship the situation is similar to the one depicted in Figs. 1a and 1b, where the complete triangle (or trapezoid) must be enumerated. On the contrary,

by considering alternating between the two relationships one can save computing time. The bivariate Poisson model arises if $\alpha_j = 0$, $j = 0, 1, 2$. Karlis (2003) examined the bivariate Charlier series distribution of Papageorgiou and Loukas (1995) which assumes that $Y_i \sim Po(\theta_i p)$, $i = 1, 2$ and $Y_0 \sim \text{Bin}(N, p)$.

Multivariate generalization of the above model is straightforward. Using similar arguments as in Hesselager (1996) one can derive similar recurrence relationships for multivariate models. Our algorithms can be used for efficient calculation of the probabilities.

A similar family of bivariate distributions is the one proposed by Wahlin and Paris (2000) with recurrence relationships of the form:

$$P(x, y) = \theta \sum_{k=1}^{\min\{x,y\}} q(k)P(x-k, y-k) + \lambda P(x-1, y)$$

and symmetric relationships with respect to y, where θ, λ are parameters and $q(k)$ a function not depending on the probabilities. One can easily see that if $q(k) = 0$ for $k > 1$ then the bivariate Poisson case is deduced.

Some known bivariate distributions with such recurrence relationships are the bivariate negative binomial distribution and the bivariate Poisson-inverse Gaussian distribution (see, also, Kocherlakota and Kocherlakota, 1992). If $q(k) \neq 0$ for $k > 1$, our algorithms still apply since the new relation just adds points at the diagonal. Again generalization to higher dimensions is straightforward. The derivation of the distributions can be represented, again, as a trivariate reduction.

For univariate distributions there are a lot of families defined via the form of their recurrence relationships. One can derive similar relationships for bivariate extensions based on trivariate reduction schemes. This is beyond the scope of the present paper.

7. CONCLUDING REMARKS

In the present paper we developed efficient algorithms for calculating multivariate probabilities for a variety of multivariate models. Among them the multivariate Poisson distribution is, perhaps, the most prominent member. It is shown that the calculation of the probabilities, even in the multivariate case are affordable, while a guidance was given about which algorithm is preferable in every situation.

The importance of such procedures for multivariate discrete distributions is obvious. For bivariate distributions a complete enumeration of the probability function might not be of special effort, but as

the dimension increases there are more cells with zero frequency and, hence, without need for evaluating the probability function at these points. So, efficient algorithms can save considerable computing time for such probabilities.

On the other hand, for a variety of models the probabilities must be calculated repetitively for different parameterizations and the complete enumeration of the probability table is totally unnecessary. We gave an example where the EM iterations need the evaluation of the probabilities in a series of points. Keeping in mind that the EM algorithm is applicable to all the cases where a trivariate reduction schemes has been used, it is obvious that efficient algorithms for the probabilities are important for speeding up the estimation task.

APPENDIX 1: THE FULL ALGORITHM

Algorithm for Stage 1

From the algorithm of stage 2 we would have given the n points on the hyperplane closest to the $\mathbf{X}$ point: $\mathbf{X} - x_1\mathbf{1}$, $\mathbf{X} - (x_1 - 1)\mathbf{1} - \mathbf{e_1}$, ..., $\mathbf{X} - (x_1 - 1)\mathbf{1} - \sum_{k=1}^{n-1} \mathbf{e_k}$.

The Algorithm proceeds as:

$$P[\mathbf{X} - (x_1 - 1)\mathbf{1}] = \theta_1 P[\mathbf{X} - (x_1 - 1)\mathbf{1} - \mathbf{e_1}] + \theta_0 P[\mathbf{X} - x_1\mathbf{1}]$$

FOR $l = 2$ TO $x_1 - 1$ (OUTER LOOP)

$$P\left[\mathbf{X} - (x_1 - l)\mathbf{1} - \sum_{k=1}^{n-1} \mathbf{e_k}\right]$$

$$= \frac{\theta_n}{x_n - (x_1 - l)} P[\mathbf{X} - (x_1 - (l-1))\mathbf{1}]$$

$$+ \frac{\theta_0}{x_n - (x_1 - l)} P\left[\mathbf{X} - (x_1 - (l-1))\mathbf{1} - \sum_{k=1}^{n-1} \mathbf{e_k}\right]$$

FOR $i = n - 2$ DOWN TO 1 (INNER LOOP)

$$P\left[\mathbf{X} - (x_1 - l)\mathbf{1} - \sum_{k=1}^{i} \mathbf{e_k}\right]$$

$$= \frac{\theta_i}{x_i - (x_1 - l)} P\left[\mathbf{X} - (x_1 - l))\mathbf{1} - \sum_{k=1}^{i+1} \mathbf{e_k}\right]$$

$$+ \frac{\theta_0}{x_i - (x_1 - l)} P\left[\mathbf{X} - (x_1 - (l-1)\mathbf{1} - \sum_{k=1}^{i} \mathbf{e_k}\right]$$

END (INNER LOOP)

$$P\left[\mathbf{X}-(x_1-l)\mathbf{1}\right] = \frac{\theta_1}{l}P\left[\mathbf{X}-(x_1-l)\mathbf{1}-\mathbf{e_1}\right]+\frac{\theta_0}{l}P\left[\mathbf{X}-(x_1-(l-1))\mathbf{1}\right]$$

END (OUTER LOOP)

$$P[\mathbf{X}] = \frac{\theta_1}{x_1}P[\mathbf{X}-\mathbf{1}]+\frac{\theta_0}{x_1}P[\mathbf{X}-\mathbf{1}-\mathbf{e_1}]$$

Algorithm for Stage 2

Calculate $P[\mathbf{0}]$

FOR $i = 1$ TO $x_n - x_{n-1}$

$$P[i \quad \mathbf{e_n}] = \frac{\theta_n}{i}P[(i-1)\mathbf{e_n}]$$

END

FOR $l = n-1$ DOWN TO 2 (OUTER LOOP)

FOR $i = 1$ TO $x_l - x_{l-1}$ (INNER LOOP)

$$P\left[i \quad \mathbf{e_l}+\sum_{k=l+1}^{n}(x_k-x_l+i)\mathbf{e_k}\right] = \frac{\theta_l}{i}\prod_{k=l+1}^{n}\frac{\theta_k}{(x_k-x_l+i)}P\left[(i-1)\mathbf{e_l}+\sum_{k=l+1}^{n}(x_k-x_l+i-1)\mathbf{e_k}\right]$$

END (INNER LOOP)

END (OUTER LOOP)

At this point we have calculated the $\mathbf{X}-x_1\mathbf{1}$ point. Then we need to calculate the rest $n-1$ points to have the complete set of points needed from the algorithm of stage 1 to run.

$$P\left[\mathbf{X}-(x_1-1)\mathbf{1}-\sum_{k=1}^{n-1}\mathbf{e_k}\right] = \frac{\theta_n}{x_n-x_1+1}P[\mathbf{X}-x_1\mathbf{1}]$$

FOR $i = n-2$ TO 1

$$P\left[\mathbf{X}-(x_1-1)\mathbf{1}-\sum_{k=1}^{i}\mathbf{e_k}\right] = \frac{\theta_{i+1}}{x_i-x_1+1}P\left[\mathbf{X}-(x_1-1)\mathbf{1}-\sum_{k=1}^{i+1}\mathbf{e_k}\right]$$

END

APPENDIX 2: THE FLAT ALGORITHM

Algorithm for Stage 1

From the algorithm of the second stage (provided in Appendix 1) we will have available the $x_1 + 1$ points on the hyperplane closest to the $\mathbf{X}$ point: $\mathbf{X} - x_1\mathbf{e_1}$, $\mathbf{X} - x_1\mathbf{e_1} - \sum_{i=2}^{n} \mathbf{e_i}$, $\mathbf{X} - x_1\mathbf{e_1} - 2\sum_{i=2}^{n} \mathbf{e_i}$, ..., $\mathbf{X} - x_1\mathbf{e_1} - x_1 \sum_{i=2}^{n} \mathbf{e_i} = \mathbf{X} - x_1\mathbf{1}$.

FOR $l = 1$ TO $x_1 - 1$ (OUTER LOOP)

FOR $m = 0$ TO $x_1 - l$ (INNER LOOP)

$$P\left[\mathbf{X} - (x_1 - l)\mathbf{e_1} - m\sum_{i=2}^{n} \mathbf{e_i}\right]$$
$$= \frac{\theta_1}{l} P\left[\mathbf{X} - (x_1 - l + 1)\mathbf{e_1} - m\sum_{i=2}^{n} \mathbf{e_i}\right]$$
$$+ \frac{\theta_0}{l} P\left[\mathbf{X} - (x_1 - l + 1)\mathbf{e_1} - (m+1)\sum_{i=2}^{n} \mathbf{e_i}\right]$$

END (INNER LOOP)

END (OUTER LOOP)

$$P[\mathbf{X}] = \frac{\theta_1}{x_1} P[\mathbf{X} - \mathbf{e_1}] + \frac{\theta_0}{x_1} P[\mathbf{X} - \mathbf{1}]$$

REFERENCES

Hesselager, O. (1996). Recursions for certain bivariate counting distributions and their compound distributions. *ASTIN Bulletin* 26:35–52.

Johnson, N. L., Kotz, S., Balakrishnan, N. (1997). *Discrete Multivariate Distributions*. New York: John Wiley.

Kano, K., Kawamura, K. (1991). On recurrence relations for the probability function of multivariate generalized poisson distribution. *Commun. Statist. Theory and Methods* 20:165–178.

Karlis, D. (2003). An EM algorithm for multivariate poisson distribution and related models. *J. Appl. Statist.* 30:63–77.

Kawamura, K. (1985). A note on the reccurence relations for the bivariate Poisson distribution. *Kodai Math. J.* 8:70–78.

Kocherlakota, S., Kocherlakota, K. (1992). *Bivariate Discrete Distributions*. New York: Marcel and Dekker.

Marcel Dekker, Inc.
270 Madison Avenue, New York, New York 10016

Krummenauer, F. (1998). Limit theorems for multivariate discrete distributions. *Metrika* 47:47–69.

Loukas, S., Kemp, C. D. (1983). On computer sampling from trivariate and multivariate discrete distributions. *J. Statist. Comput. Sim.* 17:113–123.

Mahamunulu, D. M. (1967). A note on regression in the multivariate Poisson distribution. *J. Amer. Statist. Assoc.* 62:251–258.

Mardia, K. V. (1971). *Families of Bivariate Distributions*. Griffin's Statistical Monograph No. 27, London: Charles Griffin.

Panjer, H. H. (1981). Recursive evaluation of a family of compound distributions. *ASTIN Bulletin* 12:22–26.

Papageorgiou, H., Loukas, S. (1995). A bivariate Charlier series distribution. *Biometrical J.* 37:105–117.

Wahlin, J. F., Paris, J. (2000). Recursive formulae for some bivariate counting distribution obtained by the trivariate reduction method. *ASTIN Bulletin* 30:141–155.